APPENDIX A

Heuristics Stylesheet:

```
<?xml version="1.0" encoding="utf-8" ?>

<xsl:stylesheet
 version="1.0"
 extension-element-prefixes="saxon"
 xmlns:xsl='http://www.w3.org/1999/XSL/Transform'
 xmlns:outxsl='output.xsl'
 xmlns:out='default.xsl'
>

 <xsl:output indent="yes"/>

 <!-- Declare namespace aliases.  When we write out the created
   -- stylesheet, "out" will map to the default (empty) namespace and
   -- "outxsl" will map to the "xsl" namespace.
   -->
 <xsl:namespace-alias stylesheet-prefix="out" result-prefix=""/>
 <xsl:namespace-alias stylesheet-prefix="outxsl" result-prefix="xsl"/>

 <!--
     At the beginning of the generated stylesheet, we need to emit an
     <xsl:stylesheet> and, for convenience, we declare a global variable
     to hold the representation of a newline symbol.  We also set the
     <xsl:output> to indent="yes" in the generated stylesheet to get a
     prettier output.  Then, in the generated stylesheet, emit a template
     to match on the <document> element and within that we apply-
     templates to the <line> children.
  -->
 <xsl:template match="/">
  <outxsl:stylesheet version="1.0">
```

```
<outxsl:variable name="nl" select="'&#xa;'"/>
<outxsl:output indent="yes"/>

<!-- Emit a template for <document>.  Within that template,
 -- apply-templates to the <line> children then emit a newline
 -- for prettyprinting purposes.
 -->
<outxsl:template match="document">
 <out:nitf>
 <outxsl:apply-templates select="line"/>
 <outxsl:value-of select="$nl"/>
 </out:nitf>
</outxsl:template>

<!-- Emit a template for <line>.  The xsl:apply-templates will
 -- cause the various <xsl:when> elements to be emitted.  After
 -- the <xsl:when> clauses have been emitted, we emit an
 -- <xsl:otherwise> clause that simply emits a newline character.
 -->
<outxsl:template match="line">
 <outxsl:choose>
  <xsl:apply-templates/>
  <outxsl:otherwise>
   <outxsl:value-of select="$nl"/>
   <outxsl:apply-templates/>
  </outxsl:otherwise>
 </outxsl:choose>
</outxsl:template>

 </outxsl:stylesheet>
</xsl:template>
```

```
<!-- process the headline heuristic -->
<xsl:template match="headline">
 <outxsl:when>
  <xsl:attribute name="test">
   <xsl:value-of select="concat('position()=',./line)"/>
  </xsl:attribute>
  <out:hedline>
   <outxsl:apply-templates/>
  </out:hedline>
 </outxsl:when>
</xsl:template>

<!-- process the byline heuristic -->
<xsl:template match="byline">
 <outxsl:when>
  <xsl:attribute name="test">
   <xsl:value-of select="concat('position()=',./line)"/>
  </xsl:attribute>
  <out:byline>
   <outxsl:apply-templates/>
  </out:byline>
 </outxsl:when>
</xsl:template>

<!-- process the dateline heuristic -->
<xsl:template match="dateline">
 <xsl:choose>
  <xsl:when test="./line">
   <outxsl:when>
    <xsl:attribute name="test">
     <xsl:value-of select="concat('position()=',./line)"/>
    </xsl:attribute>
```

```
<out:dateline>
  <outxsl:apply-templates/>
</out:dateline>
</outxsl:when>
</xsl:when>
<xsl:when test="./string">
  <xsl:choose>
    <xsl:when test="./string[@pos='first']">
      <outxsl:when>
        <xsl:attribute name="test">
          <xsl:value-of
            select="concat('starts-with(.,&quot;',./string,'&quot;)')"/>
        </xsl:attribute>
        <out:dateline>
          <outxsl:apply-templates/>
        </out:dateline>
      </outxsl:when>
    </xsl:when>
  </xsl:choose>
</xsl:when>
</xsl:choose>
</xsl:template>

<!-- process the paragraph heuristic -->
<xsl:template match="paragraph">
  <xsl:choose>
    <xsl:when test="blank/@type='line'">
      <outxsl:when>
        <xsl:attribute name="test">
          <xsl:text>.="</xsl:text>
        </xsl:attribute>
        <outxsl:value-of select="$nl"/>
```

```
            <out:p/>
          </outxsl:when>
        </xsl:when>
      </xsl:choose>
    </xsl:template>

</xsl:stylesheet>
```

APPENDIX B

Discrimination Stylesheet:

```
<?xml version="1.0" encoding="utf-8" ?>
<xsl:stylesheet xmlns:xsl=http://www.w3.org/1999/XSL/Transform
    version="1.0">
    <xsl:variable name="nl" select="' '" />
    <xsl:output indent="yes" />
    <xsl:template match="document">
        <nitf>
            <xsl:apply-templates select="line" />
            <xsl:value-of select="$nl" />
        </nitf>
    </xsl:template>
    <xsl:template match="line">
        <xsl:choose>
            <xsl:when test="position()=1">
                <hedline>
                    <xsl:apply-templates />
                </hedline>
            </xsl:when>
            <xsl:when test="position()=2">
                <byline>
                    <xsl:apply-templates />
                </byline>
            </xsl:when>
            <xsl:when test="starts-with(.,"[DATE]")">
                <dateline>
                    <xsl:apply-templates />
                </dateline>
            </xsl:when>
```

```
            <xsl:when test=".=''">
                <xsl:value-of select="$nl" />
                <p />
            </xsl:when>
            <xsl:otherwise>
                <xsl:value-of select="$nl" />
                <xsl:apply-templates />
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>
</xsl:stylesheet>
```